基于视觉感知特征的手机应用流量识别方法 *

李　玎，祝跃飞，林　伟

(数学工程与先进计算国家重点实验室，郑州 450001)

摘　要：由于大多数手机应用通过 HTTP 协议进行通信，传统的端口识别方法已经基本失效。另外，深度包检测和基于流统计特征的机器学习方法均存在手工设计特征和标记样本的困难。借鉴计算机视觉领域的优势，提出了一种基于视觉感知特征的手机应用流量识别方法。首先，将应用层载荷数据转换为视觉上有意义的图像，并从网络关口采集真实数据，建立了样本数据集 IMTD17；然后，设计了具有视觉特征提取能力的卷积感知网络模型 2D-CPN，利用卷积自编码实现了对大量无标记样本的学习，并通过多类型回归建立起从隐层特征到应用类型的映射。实验结果表明，该方法的流量识别准确率满足实际使用的需求。

关键词：手机应用；流量识别；卷积自编码；隐层特征

中图分类号：TP391　　doi: 10.3969/j.issn.1001-3695.2017.11.1001

Mobile app traffic identification based on visual perception features

Li Ding, Zhu Yuefei, Lin Wei

(*State Key Laboratory of Mathematical Engineering & Advanced Computing*, *Zhengzhou* 450001, *China*)

Abstract: The mobile apps mostly communicate with servers via HTTP, which makes port-based method ineffective. Furthermore, depth packet inspection and flow-based classifiers have difficulties in designing features and labeling samples manually. Motivated by the excellence of computer vision, this paper proposed a method of mobile app traffic identification based on visual perception features. First, it converted the app traffic flows into vision-meaningful images. Collecting real traffic from the network gateway, it created the IMTD17 dataset. Then, it designed a two-dimensional convolutional perception network (2D-CPN) with the ability of visual feature extraction. The network realized the learning of massive unlabeled samples by the convolutional autoencoder, and used multi-class regression to create the mapping from the latent feature to the app categories. The experimental results show that the identification accuracy of the approach satisfies the practical requirement.

Key Words: mobile app; traffic identification; convolutional autoencoder; latent feature

0　引言

流量识别（traffic identification）是将网络流量映射到网络协议或生成应用的过程，对于网络监管和恶意应用发现具有重要的作用。传统的流量识别主要针对桌面应用（desktop application），如商业软件、恶意软件等。其网络通信协议一般基于特定端口或采用含有特征字段的私有协议，因此，通过对应用层协议端口号等协议头部字段进行匹配可以达到较好的流量识别效果。

与桌面应用不同，大多数手机应用（mobile app）通过统一的 HTTP 应用层协议与服务器进行通信[1,2]，导致传统基于端口的流量识别方法基本失效。为了识别手机应用流量，一些研究者试图寻找能够唯一标志应用的指纹信息。但是手机流量中可能并不包含这种特殊的特征字段，因此寻找有效的指纹信息并且跟上手机应用的更新速度是非常困难而且耗时的。除了提取特征的困难，流量识别面临的另一个挑战是缺乏有效的数据集[3]。由于网络数据中手机应用流量规模庞大，而且手机应用本身在不断更新和产生，所以难以保证带标记流量数据集的实时性和有效性。因此，如何利用网络中大量的无标记数据是流量识别技术发展的关键。

针对上述两个挑战，本文提出了一种基于视觉感知特征的流量识别深度学习方法。首先，利用深度学习领域图像识别的思想，将手机应用流量转换为视觉上有意义的图像，使机器能够按照类似于网络分析人员使用 Wireshark 工具的方法去分析流量；然后，通过二维视觉感知网络 2D-CPN 实现对大量无标记网络流量数据的特征提取；最后，通过对多类型回归的小规模监督学习，实现对应用流量的准确识别。另外，为了测试模型的性能，同时解决数据集缺乏的难题，本文建立了手机应用

基金项目：国家自然科学基金资助项目（61271252）；国家重点研发计划资助项目（2016YFB0801601）

作者简介：李玎（1992-），男，河南郑州人，硕士，主要研究方向为机器学习、人工智能等（258166011@qq.com）；祝跃飞（1962-），男，浙江杭州人，教授，主要研究方向为信息安全、机器学习等；林伟（1986-），男，湖南常德人，讲师，主要研究方向为信息安全、大数据等.

流量样本数据集 IMTD17。

1　相关工作

目前已经有很多关于识别手机应用流量的工作，其中最直接的方法是匹配 HTTP 协议中的特殊字段。Xu 等人[4]最早使用 User-Agent 字段来识别安卓应用，但是该字段并不是强制加入应用标识的，而且手机应用开发者也并不总是遵守添加 User-Agent 字段的约定。Dai 等人[5]将请求行 URL 和 host 字段也考虑在内，并且针对不同应用建立了对应的指纹。Miskovic 等人[6]提出了名为 AppPrint 的自动化特征提取系统，能够从 HTTP 协议中综合提取 URLs 和各个头域特征，如 User-Agent 和 Cookies 等，但是该方法的识别效果依赖于人工指定的一些识别符。文献[7,8]试图将手机应用的 HTTP 聚合，认为手机用户的一次操作会产生多个 HTTP 流，如浏览网页、打开图片或视频等，并且这些数据流的相关性主要体现在数据包的长度和相邻数据包的时间间隔。尽管识别手机应用流量已经有了很多成熟的方法，但是这些方法都依赖于人工设计的特征或特征寻找方法。在这种情况下，机器无法逾越人类定义的识别边界，自动地寻找流量数据中隐藏的未知特征。

近几年来，随着计算机视觉等领域的快速发展，学术界出现了一些基于深度学习方法的流量识别技术。Wang[9]最早使用人工神经网络（artificial neural network，ANN）对 TCP 载荷数据进行学习分类。他们将带标记的 TCP 会话的前 1 024 Byte 转换为一维向量作为 ANN 模型的输入，训练完成后发现最重要的字节主要集中在开头。这种方法的局限性在于依赖于应用层协议头部的差异，对于基于同一种应用层协议的应用流量难以区分，如手机应用使用的 HTTP 协议。文献[10]借鉴了计算机视觉的方法，使用卷积神经网络（convolutional neural network，CNN）对恶意软件流量进行分类。首先将数据的前 784 个字节转换为 28×28 的二维图像，然后使用 CNN 进行监督分类学习。在选取输入数据时，该方法考虑了 TCP 流和会话的区别，并且通过实验发现包含传输层、网络层和数据链路层头部数据所达到的分类效果最好。但是这些头部数据的格式由 RFC 文档定义，字段的含义是固定明确的，通过匹配特定协议头部字段（如 TCP 协议端口、标志位等）就可以实现精准的流量识别。因此，这种数据提取方法实际上会导致信息量的损失。此外，卷积神经网络属于监督学习方法，因此同样面临难以手工标记大量网络数据样本的挑战。

2　方法和模型

本章提出了一种基于视觉感知特征的网络流量数据图像转换方法，并基于该方法建立了手机应用流量样本图像数据集 IMTD7；然后提出了用于流量识别的二维卷积感知网络模型 2D-CPN，并详细阐述了模型的推导过程。

2.1　图像转换方法

计算机视觉领域之所以取得优异的成果，一个重要原因是机器识别的图像首先是视觉可识别的有意义的图像，然后才能指导机器按照人的思路去学习更多的图像。但是网络数据本身具有结构化的特点，其数据属性和字段间的逻辑关系是比较固定的。如果将网络数据直接填充为图像，就会打破原有的结构化特点，不易被视觉识别，因此有悖于计算机视觉的本质。基于这个动机，本文尝试对网络流量数据进行格式化（但不是解析数据），然后将其转换为视觉图像。

根据 RFC 2616 规范，HTTP 协议由起始行、若干头域和正文组成。假设 R 代表请求行，S 代表状态行，H 代表头域，B 代表正文，C 代表回车换行符 CRLF，则一次由请求（Req）和应答（Res）组成的 HTTP 会话可以描述为

$$\begin{cases} Req = RC \parallel (H_{Req}C)_{1:a} \parallel C \parallel B_{Req} \\ Res = SC \parallel (H_{Res}C)_{1:b} \parallel C \parallel B_{Res} \end{cases}$$

其中：$\parallel$ 是字节连接符；a 和 b 代表对应消息中头域的个数；$(HC)_{1:x} = H_1C \parallel H_2C \parallel \ldots \parallel H_xC$。为了提取有效的 HTTP 协议字段，对各个部分的有效性进行分析。

a) 请求行 R：请求消息的首行，由方法、URL 和版本组成，其中 URL 与应用的具体请求相关。

b) 状态行 S：应答消息的首行，只包含 HTTP 状态信息，因此与应用不相关。

c) 头域 H：从 HTTP 请求和应答消息的第二行开始，其类型、数量和顺序都是不确定的。通过大量分析发现，在同一手机应用通信产生的 HTTP 会话数据中，H 的数量和顺序是基本固定的。直观的解释是：对于手机应用端，只要是公开发布的平台版本，其程序属性由开发者确定，因此 H 的特征是固定的；对于服务器端，H 的特征虽然与服务器平台有一定相关性，但主要由应用开发者对服务器的配置所决定，并且服务一般要做到版本向下兼容，因此不会频繁更新配置或加入新的特性。

d) 正文 B：请求和应答消息的最后一个部分，其数据类型多样，并且是可选项。当其内容是控制信息时，与应用的行为存在一定联系；当其内容是资源数据时，可以认为是随机的，因此与应用之间不存在紧密的相关性。

通过上述分析，选择与应用相关的 R 和 H 作为有效数据。另外，为了保留视觉可辨别的轮廓信息，保留换行符 C。因此，最终提取的有效数据为 $RC \parallel (H_{Req}C)_{1:a} \parallel C \parallel (H_{Res}C)_{1:b}$。

在提取的数据中，行数和每行的字节长度是不确定的。由于计算机视觉接受的图像范围是固定的，所以需要对提取的数据进行规范化，填充或舍弃部分数据，将其转换为可作为模型输入的样本图像。假设所要转换的目标图像大小为 $M \times N$。为了得到图像的一行数据像素点，定义行运算 P，表示为

$$P(d_{1:n}) = \begin{cases} d_{1:n} \parallel O^{N-n}, & n < N \\ d_{1:N}, & n \geq N \end{cases} \tag{1}$$

其中：$d_{1:n}$ 表示长度为 n 字节的数据片段；O^k 表示 k 个 NUL

字符的连接。为了得到完整的图像，定义列运算 Q，表示为

$$Q(e_{1:m})=\begin{cases} e_{1:m} \parallel O^{(M-m)\times N}, & m<M \\ e_{1:M}, & m\geq M \end{cases} \tag{2}$$

其中：$e_{1:m}$ 表示行数为 m 行的二维图像数据。利用式(1)和(2)，可以将提取的有效数据转换为模型输入图像（image），表示为

$$Image = Q(P(R) \parallel P(H_{Req})_{1:a} \parallel P(\varnothing) \parallel P(H_{Res})_{1:b})$$

其中：$P(H)_{1:x}=P(H_1)\parallel P(H_2)\parallel \ldots \parallel P(H_x)$；$\varnothing$ 是长度为 0 的数据。为了尽可能保留图像的视觉特征，尝试了多种图像尺寸，并确定 M=28，N=36。图 1 展示了部分手机应用流量的图像样本，其中每种应用随机选取 4 次通信产生的样本。可以看出，同一应用的不同流量样本图像具有高度的一致性，而不同应用之间的流量样本图像存在很高的差异度。因此，将该转换方法获得的图像作为流量识别模型的输入是合理的。

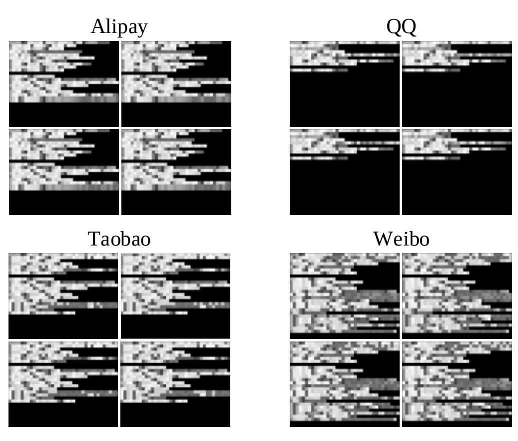


图 1　手机应用流量样本图像

基于图像转换方法，本文建立了 IMTD17 数据集。首先，从校园网关口获取流量数据，其中手机应用流量通过连入 Wi-Fi 的智能手机产生；然后，按照四元组（源 IP、源端口、目的 IP、目的端口）提取 TCP 载荷，并得到原始 HTTP 会话。IMTD17 由三个部分组成，如表 1 所示。第一部分用于第一阶段（S_1）无监督训练，由大量无标记的样本组成，其中包括少量手机应用流量样本图像（APP）和大量无关的背景流量样本图像（BG）；第二部分用于第二阶段（S_2）监督分类训练，是从第一部分中抽取的少量人工标记的手机应用流量样本，共包括 12 种分别以 0~11 标记的 Android 应用流量样本，如表 3 所示；第三部分用于模型测试（T），由带标记的样本组成，成分与第一部分相似，其中只有第二部分对应的 12 个类别样本带有真实标记，其余背景流量样本的标记值为 12。

表 1　IMTD17 包含的数据集

Dataset	Labeled	BG	APP
S_1	No	45000	5000
S_2	Yes	0	1000
T	Yes	9000	1000

2.2　卷积感知网络模型

针对 2.1 节转换得到的手机应用流量图像，本文设计了与图像尺寸相适应的二维卷积感知网络模型（2D convolutional perception network，2D-CPN），如图 2 所示。该模型基于卷积自编码算法[11,12]。相比其他自编码算法，卷积自编码能够提取原始输入局部的二维特征，并且实现权值的全局共享，而不是将所有特征视为全局特征。卷积操作保留了输入图像数据的局部相关性，将空间位置信息转换为高阶特征表示。然后，利用多层感知器（multi-layer perceptron，MLP）的非线性拟合能力，将这些高阶特征转换为低维隐层特征。由于这些隐层特征在低维空间具有可分辨的距离关系，通过 Softmax 多类型回归可以将它们映射到对应的类型，从而实现对样本类型的识别。

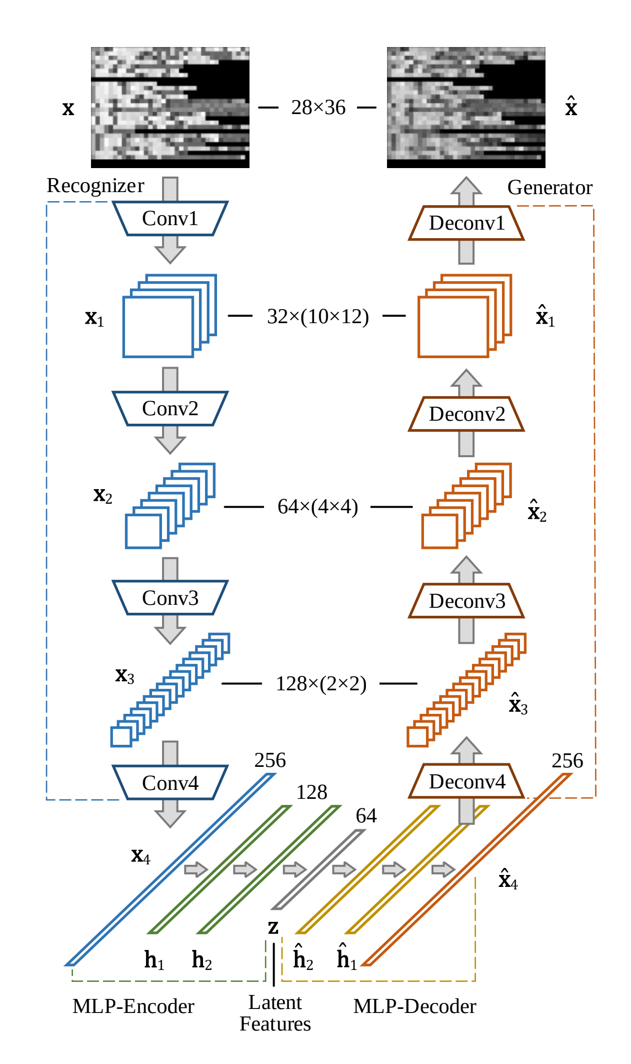


图 2　二维卷积感知网络模型 2D-CPN

在 2D-CPN 模型中，识别网络（recognizer）首先将原始 28×36 的图像样本 $\mathbf{x}$ 转换为高阶特征 $\mathbf{x}_4$，通过 MLP 编码网络（MLP-encoder）的非线性拟合，高阶特征 $\mathbf{x}_4$ 被转换为隐层特征 $\mathbf{z}$。在重建样本阶段，隐层特征 $\mathbf{z}$ 通过 MLP 解码网络（MLP-decoder）转换为对应高阶特征 $\hat{\mathbf{x}}_4$，然后生成网络（generator）将高阶特征 $\hat{\mathbf{x}}_4$ 转换为重建后的图像样本 $\hat{\mathbf{x}}$。其中，识别网络中的卷积组操作（conv）负责分解特征图，生成网络中的反卷积组操作（deconv）负责重组特征图。这两种操作中涉及到的运

算函数及其具体参数如图 3 所示。为了重建图像样本，卷积操作不应当衰减原始图像信息。然而基本的卷积操作忽视了图像的边缘特征，根据卷积核的尺寸，使输出特征图的尺寸相比输入图像有所减少。因此，需要在基本卷积操作之前加入适当的填充操作。

Conv/Deconv	1	2	3	4
h, w	3, 2	3, 2	1, 1	1, 1
k_x, k_y	5, 5	5, 5	3, 3	3, 3
s_x, s_y	3, 3	3, 3	2, 2	2, 2

图 3　2D-CPN 模型中的两种基本操作

a) Padding 函数：负责在输入图像的四周进行适当填充（padding），以扩大图像的尺寸。具体做法是根据参数 (h, w)，在输入图像的纵向和横向上分别填充对称的 h 行和 w 列内容为 0 的数据。假设 I 和 O 分别代表输入和输出图像，并且其高宽尺寸表示为 (H, W)，那么经过 padding 填充后，输出图像的尺寸变为 $H_O = H_I + 2h$，$W_O = W_I + 2w$。

b) Convolution 函数：在二维离散空间中，基本的卷积（Convolution）操作 $O = \boldsymbol{K} * I$ 可以定义为

$$O(i, j) = \sum_{u=-k_y}^{k_y} \sum_{v=-k_x}^{k_x} \boldsymbol{K}(u, v) I(i-u, j-v) \tag{3}$$

其中：$\boldsymbol{K}$ 是大小为 (k_x, k_y) 的卷积核，为了满足卷积运算的特点，k_x 和 k_y 都是奇数。经过卷积运算后，输出特征图的尺寸变为 $H_O = H_I - (k_y - 1)$，$W_O = W_I - (k_x - 1)$。经过 padding 函数填充之后，卷积运算可以产生与输入特征图尺寸相同的输出。对于图 2 识别网络的第 l 层，假设 $\mathbf{x}_{l-1} = \{\mathbf{x}_{l-1}^1, \mathbf{x}_{l-1}^2, ..., \mathbf{x}_{l-1}^M\}$ 是 $l-1$ 层中由 M 个特征图组成的输入，$\boldsymbol{K}_l = \{\boldsymbol{K}_l^1, \boldsymbol{K}_l^2, ..., \boldsymbol{K}_l^N\}$ 是 N 个卷积核，$\boldsymbol{b}_l = \{\boldsymbol{b}_l^1, \boldsymbol{b}_l^2, ..., \boldsymbol{b}_l^N\}$ 是与卷积核对应的偏置值，那么输出特征图 $\mathbf{x}_l = \{\mathbf{x}_l^1, \mathbf{x}_l^2, ..., \mathbf{x}_l^N\}$ 可以由以下运算得到

$$\mathbf{x}_l^n = \mathcal{F}(\sum_{m=1}^{M} \boldsymbol{K}_l^n * \mathbf{x}_{l-1}^m + \boldsymbol{b}_l^n) \tag{4}$$

其中：$\mathcal{F}$ 是激活函数，这里使用修正线性单元（ReLU）来加速训练时的收敛过程，同时可以避免可能的预训练[13]。

c) Subsampling 函数：下采样（subsampling）函数也称为池化操作，可以减少输入特征图中的冗余信息，提取出高阶表征。对于给定的水平和竖直参数 (s_x, s_y)，下采样函数同时在两个方向上缩减数据，得到的输出特征图尺寸为 $H_O = H_I \div s_y$，$W_O = W_I \div s_x$。

通过上述推导，图 2 识别网络中的一次卷积组操作产生的特征图尺寸为

$$\begin{cases} H_O = (H_I + 2h - (k_y - 1)) \div s_y \\ W_O = (W_I + 2w - (k_x - 1)) \div s_x \end{cases} \tag{5}$$

d) Upsampling 函数：为了重建卷积组操作前的特征图，从而最终得到重建后的输入样本。图 2 生成网络中的反卷积组操作首先进行下采样函数的逆过程，即上采样（upsampling）操作。由于 subsampling 是一种数据损失的过程，upsampling 函数要尽可能地恢复采样前的数据，所以这里不对池化后数据进行简单的重复扩展操作，而使用“‘what’ and ‘where’”反池化技术[14]来同时恢复池化前的位置和数值信息。对于给定的水平和竖直参数 (s_x, s_y)，上采样函数同时在两个方向上扩展数据，得到的输出特征图尺寸为 $H_O = H_I \times s_y$，$W_O = W_I \times s_x$。

e) Deconvolution 函数：反卷积（deconvolution）也称为转置卷积，可以看做卷积操作的相反过程[15]。对于大小为 (k_x, k_y) 的反卷积核 $\hat{\boldsymbol{K}}$，反卷积操作得到的特征图尺寸会增加，即 $H_O = H_I + (k_y - 1)$，$W_O = W_I + (k_x - 1)$。对于图 2 生成网络的第 l 层，假设 $\hat{\mathbf{x}}_{l+1} = \{\hat{\mathbf{x}}_{l+1}^1, \hat{\mathbf{x}}_{l+1}^2, ..., \hat{\mathbf{x}}_{l+1}^M\}$ 是 $l+1$ 层中由 M 个特征图组成的输入，$\hat{\boldsymbol{K}}_l = \{\hat{\boldsymbol{K}}_l^1, \hat{\boldsymbol{K}}_l^2, ..., \hat{\boldsymbol{K}}_l^N\}$ 是 N 个反卷积核，$\hat{\boldsymbol{b}}_l = \{\hat{\boldsymbol{b}}_l^1, \hat{\boldsymbol{b}}_l^2, ..., \hat{\boldsymbol{b}}_l^N\}$ 是与反卷积核对应的偏置值，并且激活函数仍然选择修正线性单元 ReLU，那么，输出特征图 $\hat{\mathbf{x}}_l = \{\hat{\mathbf{x}}_l^1, \hat{\mathbf{x}}_l^2, ..., \hat{\mathbf{x}}_l^N\}$ 可以由以下运算得到

$$\hat{\mathbf{x}}_l^n = \mathcal{F}(\sum_{m=1}^{M} (\hat{\boldsymbol{K}}_l^n)^T * \hat{\mathbf{x}}_{l+1}^m + \hat{\boldsymbol{b}}_l^n) \tag{6}$$

f) Cropping 函数：作为 padding 函数的逆运算，cropping 函数的作用是对输入图像的边缘进行适当的裁剪。具体的做法是根据参数 (h, w)，对输入图像的纵向和横向分别裁剪对称的 h 行和 w 列数据。经过 cropping 函数裁剪后，输出图像的尺寸变为 $H_O = H_I - 2h$，$W_O = W_I - 2w$。

通过上述推导，图 2 生成网络中的一次反卷积组操作产生的特征图尺寸为

$$\begin{cases} H_O = H_I \times s_y + (k_y - 1) - 2h \\ W_O = W_I \times s_x + (k_x - 1) - 2w \end{cases} \tag{7}$$

图 2 生成网络将原始图像 **x** 转换为具有良好表示的高阶特征 $\mathbf{x}_4$。为了实现分类识别，使用 MLP 对高阶特征进行非线性拟合，从而得到从高阶特征到隐层特征 **z** 的映射。对于图 2 中的 MLP 编码网络和解码网络，其第 *l* 层可以分别表示为

$$\mathbf{h}_l = \mathcal{F}(\boldsymbol{W}_l \mathbf{h}_{l-1} + \boldsymbol{b}_l) \tag{8}$$

$$\hat{\mathbf{h}}_l = \mathcal{F}(\hat{\boldsymbol{W}}_l \hat{\mathbf{h}}_{l+1} + \hat{\boldsymbol{b}}_l) \tag{9}$$

其中：$\boldsymbol{W}$ 和 $\boldsymbol{b}$ 是感知器节点权重和偏置值；$\mathcal{F}$ 是各层的激活函数，这里使用 softplus 以更好地将高阶特征均匀映射到隐层空间[16]。假设多层感知器的层数为 *p*（通常令 *p*=2 以避免过拟合现象[17]），那么隐层特征 **z** 通过 MLP 编码网络得到，即

$$\mathbf{z} = \boldsymbol{W}_z \mathbf{h}_p + \boldsymbol{b}_z \tag{10}$$

当进行分类识别时，模型通过多类型回归 softmax 函数将隐层特征 **z** 映射到某个输出类别。Softmax 层节点的输出值表示原始输入样本属于对应类型的可信度，即

$$P(\hat{\mathbf{y}})_j = e^{\hat{y}_j} \Big/ \sum_{i=1}^{\mathrm{Dim}(\mathbf{y})} (e^{\hat{y}_i}) \tag{11}$$

模型涉及的训练和测试过程如图 4 所示。在正式训练之前，图 4(a)和(b)两个预训练过程可以帮助更加快速准确地提取隐层特征 **z**。正式训练过程由图 4(c)无监督重建模型（reconstructor）训练和(d)监督分类模型（categorizer）训练两个阶段组成。其中，前一阶段类似于婴儿观察客观世界的过程，它们通过观察能够区分不同类别的事物，但是不知道具体是什么事物，而后一阶段相当于教导婴儿他们所区分的事物是什么。

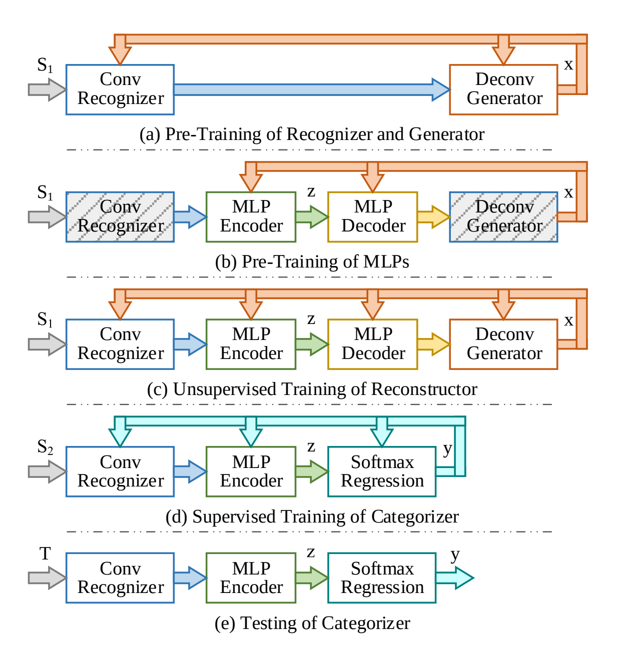


图 4　2D-CPN 模型的训练和测试过程

在无监督训练阶段，重建模型从无标记数据集 S_1 中提取隐层特征 **z**。训练开始前，各卷积核和多层感知器权重使用 Xavier 方法初始化，使各层输出的方差尽可能相等[18]。为了通过反向传播算法得到良好的参数，训练时重建模型使用交叉熵作为损失函数，即

$$C(\mathbf{x}) = -\sum_{i=1}^{\mathrm{Dim}(\mathbf{x})} (\mathrm{x}_i \log \hat{\mathrm{x}}_i + (1-\mathrm{x}_i)\log(1-\hat{\mathrm{x}}_i)) \tag{12}$$

同时，各卷积核、多层感知器权重和偏置值通过反向传播算法进行修正。在监督训练阶段，分类模型通过标记数据集 S_2 建立从隐层特征 **z** 到特定类别的映射。训练时，分类模型使用多类别交叉熵损失函数，即

$$C(\mathbf{y}) = -\sum_{i=1}^{\mathrm{Dim}(\mathbf{y})} (\mathrm{y}_i \log \hat{\mathrm{y}}_i + (1-\mathrm{y}_i)\log(1-\hat{\mathrm{y}}_i)) \tag{13}$$

其中：**y** 为 one-hot 编码的类型向量矩阵；$\hat{\mathbf{y}}$ 为 Softmax 层节点输出向量矩阵。值得注意的是，实际网络中大多数流量数据都是未知类别的背景流量。然而 softmax 层输出的类型结果具有明显的边界限制，即 dim(**y**)。如果只根据输出的可信值进行分类，就会产生很多错误的分类结果。因此，对 softmax 层输出结果加入阈值条件，只有当可信值大于对应阈值时，该样本才认为属于对应类型。

3　实验及分析

本章通过对 2D-CPN 模型的训练和测试过程进行实验与分析，首先验证模型提取样本隐层特征的能力，然后利用测试数据集测试模型识别流量的性能。

3.1　隐层特征可视化

为了说明利用隐层特征可以进行分类识别，从而实现对产生流量应用的识别，在二维空间对隐层特征进行可视化。具体的做法是：首先降低隐层空间的维度，令 dim(**z**)=2，使隐层特征被映射到平面上的点；然后重新训练图 4(c)重建模型，建立从输入图像样本 **x** 到二维隐层特征 **z** 的映射；训练完成后，使用包含应用流量和大量背景流量的测试数据集 T 作为重建模型的输入，得到隐层特征 **z**；最后将数据集 T 中样本的标记值 **y**（表 3）与隐层特征 **z** 相对应，以隐层特征 **z** 的二维数值作为平面的坐标点，绘制应用流量和背景流量样本的位置，如图 5 所示。为了方便区分，不同的应用流量样本点使用不同的形状表示，背景流量样本点使用小圆圈表示。

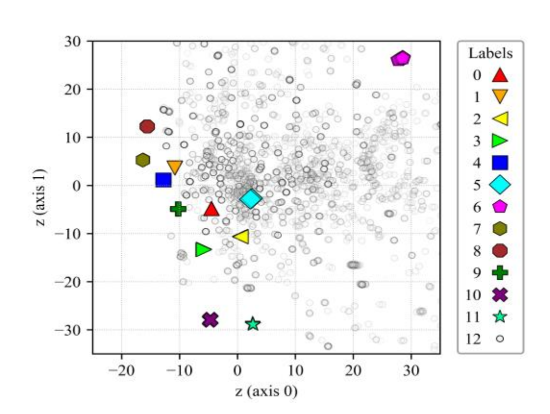


图 5　数据集 T 中样本在二维隐层特征分布

从图中可以看出，同一类样本的隐层特征非常集中，说明模型提取的隐层特征具有高度的统一性；不同类别样本的隐层特征之间具有足够的区分距离，说明模型提取的隐层特征具有高度的区分性。另外，背景流量样本均匀地分布在坐标原点附近，值得关注的是其中一些集中的样本点，它们很有可能属于同一种流量类别。因此，这种方法能够帮助人们发现未知类别的应用流量。通过二维可视化分析，可以对隐层特征的表示能力进行合理的推断：随着隐层空间维度的增加，不同类别样本的隐层特征之间的空间距离进一步扩大，从而隐层特征的样本表示能力进一步增强。

3.2 流量识别

图 4(e)的流量识别过程基于训练好的分类器，通过测试数据集 T 测试模型的识别效果。分类模型的识别结果通过准确率（*Acc*）、精确率（*Pre*）和查全率（*Rec*）这三个评价标准来进行评估，分别表示为

$$Acc = \frac{TP+TN}{TP+TN+FP+FN} \times 100\% \tag{14}$$

$$Pre = \frac{TP}{TP+FP} \times 100\% \tag{15}$$

$$Rec = \frac{TP}{TP+FN} \times 100\% \tag{16}$$

其中：T 和 F 分别代表正确和错误的分类数；P 和 N 分别代表正实例和负实例数，分别对应本文中的手机应用流量样本数和未知背景流量样本数。在这些评价指标中，准确率反映模型总体分类性能；精确率是针对预测结果而言的，表示预测为正的样本中有多少是真正的正样本；查全率是针对原来的样本而言的，表示样本中的正例有多少被正确预测了。

当各类别的阈值达到最佳阈值时，流量识别结果达到最优，如表 2 所示。可以看出，虽然识别的正确率和准确率较高，但是由于一些混淆背景流量的存在，导致查全率较低。

表 2 模型对数据集 T 的识别结果

Metric	*Acc*	*Pre*	*Rec*
Value	99.5	100	95.4

为了找出具体是哪些背景流量影响识别结果，从而对这些影响因素做进一步研究，分别列出每个类别手机应用的识别结果（在这种情况下，负样本数 N 代表背景流量样本数加上无关手机应用流量样本数），如表 3 所示。识别结果中只有个别应用的查全率较低，如 QQ Mail 和 WeChat，说明一些背景流量图像与这些应用流量图像非常相似，并且进入了这些应用流量样本隐层特征的识别边界。然而从这些错误的识别结果中可以获得有借鉴意义的指导，一方面，这些结果可以帮助人们检查手工标记是否正确；另一方面，这种方法可以自动化地发现未知流量类型，从而帮助人们实现更加智能的流量识别。

表 3 各类型手机应用标记及识别结果

Label	App	*Pre*	*Rec*	Label	App	*Pre*	*Rec*
0	Alipay	100	100	6	QQ	100	100
1	Baidu	100	100	7	QQ Mail	100	81.9
2	Bilibili	100	91.2	8	QQ Music	100	100
3	CNTV	100	95.7	9	Taobao	100	100
4	JD	100	100	10	WeChat	100	87.7
5	Kugou	100	100	11	Weibo	100	100

4 结束语

本文提出了一种基于视觉感知特征的手机应用流量识别方法。借鉴计算机视觉的优势，将原始应用层载荷数据转换为视觉上有意义的图像。同时，利用卷积自编码算法的视觉特征提取能力，保留了输入图像的局部相关性，实现特征权值的全局共享。最后，利用卷积感知网络实现了对大量无标记样本的自动学习以及监督下的流量识别。结果表明，模型对流量样本有很好的适应性，识别精确度达到了实际使用的需求。

参考文献：

[1] Falaki H, Lymberopoulos D, Mahajan R, et al. A first look at traffic on smartphones [C]// Proc of ACM SIGCOMM Conference on Internet Measurement. 2010: 281-287.

[2] Lee S W, Park J S, Lee H S, et al. A study on smart-phone traffic analysis [C]// Proc of IEEE Network Operations and Management Symposium. 2011: 1-7.

[3] Dainotti A, Pescape A, Claffy K C. Issues and future directions in traffic classification [J]. Network IEEE, 2012, 26 (1): 35-40.

[4] Xu Q, Erman J, Gerber A, et al. Identifying diverse usage behaviors of smartphone apps [C]// Proc of ACM SIGCOMM Conference on Internet Measurement Conference. 2011: 329-344.

[5] Dai Shuaifu, Tongaonkar A, Wang Xiaoyin, et al. NetworkProfiler: towards automatic fingerprinting of Android apps [C]//Proc of the 32nd IEEE Conference on Compute Communications.2005:809-817

[6] Miskovic S, Lee G M, Liao Y, et al. AppPrint: automatic fingerprinting of mobile applications in network traffic [M]// Passive and Active Measurement. Springer International Publishing. 2015: 57-69.

[7] Mongkolluksamee S, Visoottiviseth V, Fukuda K. Enhancing the performance of mobile traffic identification with communication patterns [C]// Proc of IEEE Computer Software and Applications Conference. 2015: 336-345.

[8] Su X, Zhang D, Dai S, et al. Mobile traffic identification based on application's network signature [J]. International Journal of Embedded Systems, 2016, 8 (2//3): 217.

[9] Wang Z. The applications of deep learning on traffic identification [EB/OL]. (2015) [2017-10-19]. http://www. blackhat. com/docs/us-15/materials/us-

15-Wang-The-Applications-Of-Deep-Learning-On-Traffic-Identification-wp. pdf.

[10] Wang W, Zhu M, Zeng X, et al. Malware traffic classification using convolutional neural network for representation learning [C]// Proc of IEEE International Conference on Information Networking. 2017: 712-717.

[11] Masci J, Meier U. Stacked convolutional auto-encoders for hierarchical feature extraction [C]// Proc of International Conference on Artificial Neural Networks. [S. l.] : Springer-Verlag, 2011: 52-59.

[12] Zeiler M D, Taylor G W, Fergus R. Adaptive deconvolutional networks for mid and high level feature learning [C]// Proc of International Conference on Computer Vision. [S. l.] : IEEE Computer Society, 2011: 2018-2025.

[13] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks [C]// Proc of International Conference on Neural Information Processing Systems. [S. l.] : Curran Associates Inc. 2012: 1097-1105.

[14] Zhao J, Mathieu M, Goroshin R, et al. Stacked what-where auto-encoders [J]. Computer Science, 2015, 15 (1): 3563-3593.

[15] Dosovitskiy A, Springenberg J, Tatarchenko M, et al. Learning to generate chairs, tables and cars with convolutional networks [J]. IEEE Trans on Pattern Analysis & Machine Intelligence, 2017, 39 (4): 692.

[16] Glorot X, Bordes A, Bengio Y. Deep sparse rectifier neural networks [C]// Proc of International Conference on Artificial Intelligence and Statistics. 2012: 315-323.

[17] Lawrence S, Giles C L. Overfitting and neural networks: conjugate gradient and backpropagation [C]// Proc of Ieee-Inns-Enns International Joint Conference on Neural Networks. 2002: 114-119 vol. 1.

[18] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks [J]. Journal of Machine Learning Research, 2010, 9: 249-256.